AIAA 2013-2568

21st AIAA Computational Fluid Dynamics Conference, 24 - 27 June, San Diego, California, 2013

# First, Second, and Third Order Finite-Volume Schemes for Advection-Diffusion

Hiroaki Nishikawa*

*National Institute of Aerospace, Hampton, VA 23666, USA*

**In this paper, we present implicit first-, second-, and third-order finite-volume schemes for advection-diffusion problems based on the method of hyperbolic systems. In particular, we demonstrate that the construction of an uniformly accurate third-order advection-diffusion scheme is made trivial by the hyperbolic method while a naive construction of adding a third-order diffusion scheme to a third-order advection scheme can fail to yield third-order accuracy. We demonstrate also that the gradients are computed simultaneously to the same order of accuracy as that of the solution variable on irregular triangular grids: first, second and third order accurate gradients by the developed first, second, and third order schemes, respectively. Furthermore, the first and second order schemes are shown to achieve one order higher accuracy for the solution variable in the advection limit.**

## I. Introduction

This paper is a sequel to the paper[1] presented at 51st AIAA Aerospace Sciences Meeting held in January 2013. In the previous paper, we presented the construction of first, second, and third order diffusion schemes by the method introduced in Ref.2. In this method, diffusion schemes are constructed from advection schemes via an equivalent hyperbolic system. We demonstrated that first, second, and third order diffusion schemes constructed by the upwind flux yield first, second, and third order accurate solution and gradients, respectively, on irregular triangular grids, with orders-of-magnitude acceleration in convergence over a traditional scheme. In particular, the third-order diffusion scheme was shown to be incomparably more accurate and efficient, providing third-order accuracy in the solution as well as in the gradients nearly at the cost of the second-order edge-based finite-volume scheme.

Towards the goal of developing a robust, accurate, and efficient three-dimensional viscous solver capable of producing high accurate derivatives (e.g, viscous stresses, heat fluxes, and vorticity) on unstructured grids, we now consider the advection-diffusion equation. The main focus of the paper is on the uniform third-order accuracy from the advection limit to the diffusion limit. The third-order scheme considered in the current study belongs to the class of numerical schemes based on vanishing residuals.[3,4,5] The second-order error term contains the residual that vanishes in the steady state; the leading error is then upgraded to third order. This type of scheme is known to deliver high-order accuracy on a relatively compact stencil (e.g., third-order accuracy on a second-order stencil), and thus very attractive for practical computations where high accuracy is demanded at a minimal additional cost. Successful applications of these schemes to the advection-diffusion equation require a careful construction to ensure the property of vanishing residuals. If not designed properly, the scheme loses the design accuracy at least by one order as shown in Ref.4 for a similar high-order scheme, and in Ref.6 for conservation laws with source terms. To achieve third-order accuracy for the advection-diffusion equation, the advective and diffusive terms must be discretized not only to third order but also in a compatible manner, and it is not trivial.

A radical approach to ensure the uniform accuracy is to integrate the advective and diffusive terms into a single hyperbolic system as proposed in Ref.7. In this approach, the accuracy degradation cannot occur because there is only a single hyperbolic system. In this paper, towards the extension to the compressible Navier-Stokes equations for which the full integration of the inviscid term and the viscous term remains a challenge, we consider a simplified approach. We construct a third-order advection-diffusion scheme as a sum of a third-order advection scheme and a third-order *hyperbolic* diffusion scheme developed in the previous paper.[1]

*Senior Research Scientist (hiro@nianet.org), National Institute of Aerospace, 100 Exploration Way, Hampton, VA 23666 USA

Such an approach often fails for conventional methods, but it works trivially for the diffusion scheme based on the hyperbolic diffusion system because both advective and diffusive terms are discretized by the same scheme. This approach is directly applicable to the compressible Navier-Stokes equations because the viscous terms can be written alone as a single hyperbolic system and the eigen-structure can be fully analyzed as shown in Ref.8.

The paper begins with the hyperbolic formulation for the advection-diffusion system, describes implicit first, second, and third order node-centered edge-based finite-volume schemes, presents numerical results and discussion, and concludes with remarks.

## II. Hyperbolic Advection-Diffusion System

Consider the advection-diffusion equation in two dimensions:

$$\partial_t u + a\,\partial_x u + b\,\partial_y u = \nu\,(\partial_{xx} u + \partial_{yy} u), \tag{II.1}$$

where $u$ is the solution variable, $(a, b)$ is a constant advection vector, and $\nu$ is a constant diffusion coefficient. Construction of numerical schemes for the advective term is relatively straightforward, but the same is not necessarily true for the diffusive term of second derivatives, especially for high-order methods and unstructured grids. A radical approach to diffusion is to convert the diffusive term into a hyperbolic system,[1] which is extended to the advection-diffusion in the following form:

$$\begin{aligned} \partial_t u + a\,\partial_x u + b\,\partial_y u &= \nu\,(\partial_x p + \partial_y q), \\ \partial_t p &= (\partial_x u - p)/T_r, \\ \partial_t q &= (\partial_y u - q)/T_r, \end{aligned} \tag{II.2}$$

where $p$ and $q$ are called the gradient variables which relax to the solution derivatives, $\partial_x u$ and $\partial_y u$, respectively, in the steady state, and $T_r$ is a *free parameter* called the relaxation time. The system is equivalent to the advection-diffusion equation (II.1) in the steady state for any $T_r$. Therefore, the steady solution to the advection-diffusion equation (II.1) can be computed by solving the first-order system. Discretization is made simple because there are no second derivatives and the system is *hyperbolic*[7] for which a variety of well established techniques are available. The hyperbolic system here designed specifically for steady computations is simply called the hyperbolic advection-diffusion system. Time-accurate computations are possible by implicit time stepping schemes, but it is beyond the scope of the present paper. This formulation clearly shows that the method is different from other relaxation models.[9,10] Our method introduces relaxation only on the diffusive fluxes, and our target equation is exactly the advection-diffusion equation (II.1), not an asymptotic approximation. Also, our target applications are second- or higher-order partial differential equations, such as the Navier-Stokes equations, not specific to rarefied gas dynamics or radiation hydrodynamics. Furthermore, the relaxation is not stiff at all because $T_r$ is a free parameter and does not have to be small. The method recovers the target equation exactly in the steady state for any finite $T_r$.

Write the system in the vector form,

$$\partial_t \mathbf{U} + \partial_x \mathbf{F} + \partial_y \mathbf{G} = \mathbf{S}, \tag{II.3}$$

where

$$\mathbf{U} = \begin{bmatrix} u \\ p \\ q \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} au - \nu p \\ -u/T_r \\ 0 \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} bu - \nu q \\ 0 \\ -u/T_r \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} 0 \\ -p/T_r \\ -q/T_r \end{bmatrix}. \tag{II.4}$$

In Ref.7, the system is taken as a single hyperbolic system. In this paper, we consider the advective term and the diffusive term separately.

$$\mathbf{F} = \mathbf{F}^a + \mathbf{F}^d = \begin{bmatrix} au \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -\nu p \\ -u/T_r \\ 0 \end{bmatrix}, \quad \mathbf{G} = \mathbf{G}^a + \mathbf{G}^d = \begin{bmatrix} bu \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -\nu q \\ 0 \\ -u/T_r \end{bmatrix}. \tag{II.5}$$

The flux Jacobian projected along an arbitrary vector, $\mathbf{n} = (n_x, n_y)$ is given by

$$\mathbf{A}_n = \frac{\partial \mathbf{F}}{\partial \mathbf{U}} n_x + \frac{\partial \mathbf{G}}{\partial \mathbf{U}} n_y = \mathbf{A}_n^a + \mathbf{A}_n^d, \tag{II.6}$$

where $\mathbf{A}_n^a$ and $\mathbf{A}_n^d$ are the advective and diffusive Jacobians, respectively,

$$\mathbf{A}_n^a = \frac{\partial \mathbf{F}^a}{\partial \mathbf{U}} n_x + \frac{\partial \mathbf{G}^a}{\partial \mathbf{U}} n_y = \begin{bmatrix} a_n & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{A}_n^d = \frac{\partial \mathbf{F}^d}{\partial \mathbf{U}} n_x + \frac{\partial \mathbf{G}^d}{\partial \mathbf{U}} n_y = \begin{bmatrix} 0 & -\nu n_x & -\nu n_y \\ -n_x/T_r & 0 & 0 \\ -n_y/T_r & 0 & 0 \end{bmatrix}, \tag{II.7}$$

and $a_n = a n_x + b n_y$. The advective Jacobian has the eigenvalue, $a_n$, and the diffusive Jacobian has the following eigenvalues:

$$\lambda = \pm\sqrt{\frac{\nu}{T_r}}, \quad 0. \tag{II.8}$$

The zero eigenvalue corresponds to the inconsistency damping mode,[2] acting on the components of $p$ and $q$ such that $q_x - p_y \neq 0$. It can be made nonzero by the fully hyperbolic formulation introduced in Ref.1, which we shall employ in this study for constructing a third-order scheme. We treat the diffusive part independently and thus define the relaxation time as suggested for pure diffusion in the previous paper:

$$T_r = \frac{L_r^2}{\nu}, \quad Lr = \frac{1}{2\pi}. \tag{II.9}$$

With the purely diffusive relaxation time, the system reduces to the scalar advection equation in the advection limit, $\nu \to 0$:

$$\begin{aligned} \partial_t u + a\,\partial_x u + b\,\partial_y u &= 0, \\ \partial_t p &= 0, \\ \partial_t q &= 0, \end{aligned} \tag{II.10}$$

whereas $T_r$ considered in Ref.7 generates a coupled system. The decoupling is suitable for modeling the hyperbolic Navier-Stokes system[8] where the viscous stresses and heat fluxes are taken as the gradient variables, which are not physically coupled with the inviscid terms. In the inviscid limit, the hyperbolic Navier-Stokes system proposed in Ref.8 reduces exactly to the Euler equations. Also, the separate treatment of the inviscid and viscous terms dramatically simplifies the construction of numerical schemes because it only requires the eigen-structure of each term, which can be fully analyzed independently. In other words, the inviscid scheme can be chosen independently from the choice of the hyperbolic viscous scheme. This simplified approach was first considered for the hyperbolic Navier-Stokes system in Ref.8, but not studied for the model equation before. This paper only considers the linear model equation, but the method can be extended to nonlinear equations by the local-preconditioning formulation proposed in Ref.8.

## III. Node-Centered Edge-Based Finite-Volume Scheme

### III.A. Discretization

The node-centered edge-based finite-volume scheme for Equation (II.3) is given by

$$V_j \frac{d\mathbf{U}_j}{dt} = -\sum_{k \in \{k_j\}} \Phi_{jk} A_{jk} + \mathbf{S}_j V_j, \tag{III.1}$$

where $V_j$ is the measure of the dual control volume around node $j$ in the set $\{J\}$ of nodes, $\{k_j\}$ is a set of neighbors of $j$, $\Phi_{jk}$ is a numerical flux, and $A_{jk}$ is the magnitude of the directed area vector, i.e., $A_{jk} = |\mathbf{n}_{jk}| = |\mathbf{n}_{jk}^\ell + \mathbf{n}_{jk}^r|$ (see Figure 1). This formulation is valid for triangular, quadrilateral, or mixed grids, and all schemes developed below can be directly applied to any grid except the third-order scheme, which is third-order accurate only

on triangular grids. For the third-order scheme, the point-source integration in Equation (III.1) cannot be employed; it needs to be discretized carefully to preserve the accuracy as will be discussed later. Note also that an appropriate boundary flux must be supplied at the boundary node. For first-order schemes, a point evaluation is sufficiently accurate, but for second-order schemes, a different quadrature is required for the linear exactness in the flux integration. See Ref.11 for a comprehensive list of linearity preserving boundary quadrature formulas.

### III.B. Numerical Flux

The numerical flux is computed by the upwind flux:

$$\Phi_{jk} = \frac{1}{2}(\mathbf{H}_L + \mathbf{H}_R)\cdot\hat{\mathbf{n}}_{jk} - \frac{1}{2}|\mathbf{A}_n|(\mathbf{U}_R - \mathbf{U}_L), \tag{III.2}$$

where $\mathbf{H}_L = [\mathbf{F}_L, \mathbf{G}_L]$, $\mathbf{H}_R = [\mathbf{F}_R, \mathbf{G}_R]$, and $\hat{\mathbf{n}}_{jk} = (n_x, n_y)$ is the unit directed area vector. The left and right fluxes and solutions are defined at the edge midpoint and evaluated by the nodal values for first-order accuracy and by the linear extrapolation from the nodes for second/third-order accuracy. The absolute Jacobian, $|\mathbf{A}_n|$, requires the full eigen-structure of the target system. The eigen-structure of the hyperbolic advection-diffusion system (II.3) is simple enough to enable the construction of the upwind flux.[7] In this paper, however, we consider a simplified construction, which was proposed for the hyperbolic Navier-Stokes scheme in Ref.8. In this approach, the numerical flux is defined by the sum of the upwind advection flux and the upwind hyperbolic-diffusion flux, which can be written as

$$\Phi_{jk} = \frac{1}{2}(\mathbf{H}_L + \mathbf{H}_R)\cdot\hat{\mathbf{n}}_{jk} - \frac{1}{2}\left(|\mathbf{A}_n^a| + |\mathbf{A}_n^d|\right)(\mathbf{U}_R - \mathbf{U}_L), \tag{III.3}$$

where

$$|\mathbf{A}_n^a| = \begin{bmatrix} |a_n| & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad |\mathbf{A}_n^d| = \frac{\nu}{L_r}\begin{bmatrix} 1 & 0 & 0 \\ 0 & n_x^2 & n_x n_y \\ 0 & n_x n_y & n_y^2 \end{bmatrix}. \tag{III.4}$$

In this way, each absolute Jacobian can be constructed independently for the advective term and the diffusive term. It has been successfully applied to the hyperbolic Navier-Stokes system:[8] the inviscid term and the viscous terms are both hyperbolic and their eigen-structures can be fully analyzed independently.

### III.C. Implicit Solver

A steady-state solution can be obtained by marching in time towards the steady state as demonstrated in the previous paper.[1] In this paper, we drop the time derivative term,

$$0 = -\sum_{k\in\{k_j\}} \Phi_{jk}A_{jk} + \mathbf{S}_j V_j, \tag{III.5}$$

and construct an implicit solver for the global system of steady residual equations. The advantage of $O(1/h)$ acceleration in the steady convergence over traditional methods, which has been observed for explicit time-marching schemes,[2,7,8,1] now comes in the iterative convergence of the linear system arising from the implicit formulation as will be demonstrated later.

Consider the global system of residual equations, which consists of rows of the nodal residual (III.5):

$$0 = \mathbf{Res}(\mathbf{U}_h), \tag{III.6}$$

where $\mathbf{U}_h$ denotes the global solution vector for which the system is to be solved. We consider the iterative method in the form:

$$\mathbf{U}_h^{n+1} = \mathbf{U}_h^n + \Delta\mathbf{U}_h, \tag{III.7}$$

where the correction $\Delta\mathbf{U}_h$ is the solution to the following linear system:

$$\frac{\partial\mathbf{Res}}{\partial\mathbf{U}_h}\Delta\mathbf{U}_h = -\mathbf{Res}(\mathbf{U}_h^n). \tag{III.8}$$

The Jacobian matrix is constructed by differentiating the residual of the first-order scheme for all schemes. Therefore, the method is Newton's method (exact for linear problems) for the first-order scheme, and a defect correction method for the second and third order schemes, provided the linear system is fully solved. In practice, we do not fully solve but relax the linear system. In this work, we employ the sequential Gauss-Seidel (GS) relaxation to relax the linear system to a specified tolerance. It is possible to add a pseudo-time term to the left hand side, but it is not used in this work. It is emphasized that the condition number of the Jacobian is $O(1/h)$, not $O(1/h^2)$ even in the diffusion limit, implying $O(1/h)$ acceleration in convergence over traditional schemes for diffusion dominated problems.

### III.D. First-Order Scheme

The simplest way to construct a first-order scheme is to evaluate the left and right states by the nodal solutions:

$$\mathbf{U}_L = \mathbf{U}_j, \qquad \mathbf{U}_R = \mathbf{U}_k, \tag{III.9}$$

and the numerical flux by the upwind flux (III.3). The resulting scheme corresponds to Scheme I in Ref.1. In this paper, we employ a modified version of Scheme I called Scheme II where the gradient variables are used to enable the linear extrapolation of the solution:

$$u_L = u_j + \frac{1}{2}\left(p_j, q_j\right) \cdot \Delta\mathbf{l}_{jk}, \qquad u_R = u_k - \frac{1}{2}\left(p_k, q_k\right) \cdot \Delta\mathbf{l}_{jk}, \tag{III.10}$$

where $\Delta\mathbf{l}_{jk} = (x_k - x_j, y_k - y_j)$. The scheme remains compact and first-order accurate in the diffusion limit as shown in the previous paper.[1] The Jacobian is constructed exactly based on this scheme. In this paper, we demonstrate that the scheme can achieve second-order accuracy in the advection limit.

### III.E. Second-Order Scheme

For second-order accuracy, we compute the nodal gradient by a linear least-squares (LSQ) method, and evaluate the left and right states by the linear extrapolation from the nodes. Again, we employ Scheme II and avoid the gradient computation for the solution by using $(p, q)$ as in Equation (III.10). The gradient computation is required only for $p$ and $q$ to perform the linear extrapolation:

$$p_L = p_j + \frac{1}{2}\nabla p_j \cdot \Delta\mathbf{l}_{jk}, \qquad p_R = p_k - \frac{1}{2}\nabla p_k \cdot \Delta\mathbf{l}_{jk}, \tag{III.11}$$

$$q_L = q_j + \frac{1}{2}\nabla q_j \cdot \Delta\mathbf{l}_{jk}, \qquad q_R = q_k - \frac{1}{2}\nabla q_k \cdot \Delta\mathbf{l}_{jk}, \tag{III.12}$$

where $\nabla p_j$ is the gradient of $p$ computed by the LSQ method at $j$, and similarly for $\nabla q_j$ . The numerical flux is computed by the upwind flux (III.3). This scheme is known to be second-order accurate for both the solution and the gradients in the diffusion limit.[1] On the other hand, the scheme produces third-order accurate solutions in the advection limit as shown later.

### III.F. Third-Order Scheme

As in the previous work, we consider the third-order edge-based finite-volume scheme discovered by Katz and Sankaran.[12] It is a very special node-centered scheme for hyperbolic conservation laws: the second-order node-centered edge-based finite-volume scheme achieves third-order accuracy on triangular grids if the nodal gradients are exact for quadratic functions and the flux is linearly extrapolated to the edge-midpoint in the case of nonlinear fluxes. The third-order accuracy has been demonstrated for regular as well as irregular triangular grids in Refs.12, 13, 14. It is a very economical third-order scheme: third-order accuracy obtained nearly at the cost of the second-order edge-based finite-volume scheme. Nevertheless, its extensions to other types of equations including source terms are not straightforward. As the scheme relies on the second-order error term that vanishes in the steady state, every term in a target equation must be discretized with a compatible second-order error term. For source terms, a systematic method to ensure the compatible discretizaton has been devised in Ref.6. For the diffusive term in the original form (II.1), a compatible third-order scheme has yet to be discovered to the author's knowledge. In this section, we first illustrate the compatibility problem for the advection-diffusion equation, and then present the hyperbolic construction as a radically simple way to avoid the problem.

*III.F.1. Third-Order Accuracy for Advection-Diffusion*

Consider the advection-diffusion equation,

$$\partial_x f + \partial_y g = \nu\,(\partial_{xx} u + \partial_{yy} u), \tag{III.13}$$

where $(f, g) = (au, bu)$. In the absence of the diffusive term (e.g., $\nu = 0$), the third-order scheme has the following local truncation error at node $j$ on a regular triangular grid:[6]

$$\mathcal{T}_j^{\text{adv}} = C_1\,\partial_{xx}(\partial_x f + \partial_y g) + C_2\,\partial_{xy}(\partial_x f + \partial_y g) + C_3\,\partial_{yy}(\partial_x f + \partial_y g) + O(h^3), \tag{III.14}$$

where the derivatives are defined at $j$, $h$ is a typical mesh spacing, and the coefficients, $C_1$, $C_2$, and $C_3$, are geometrical constants of $O(h^2)$. Without loss of generality, we focus on a regular triangular grid composed of isosceles right triangles of spacing $h$ where the truncation error is specifically given by

$$\mathcal{T}_j^{\text{adv}} = \frac{h^2}{12}\,[\partial_{xx}(\partial_x f + \partial_y g) + \partial_{xy}(\partial_x f + \partial_y g) + \partial_{yy}(\partial_x f + \partial_y g)] + O(h^3). \tag{III.15}$$

The second-order error term will vanish because $\partial_x f + \partial_y g = 0$ for the exact solution or equivalently because $\partial_x f + \partial_y g = 0$ in the steady state, and thus the truncation error is upgraded to third-order. Consequently, the discretization error is expected to be third-order. In order to achieve third order accuracy for the advection-diffusion equation, the diffusion scheme must have a second-order error term in the form:

$$\mathcal{T}_j^{\text{diff}} = -\frac{h^2}{12}\,[\partial_{xx}(\nu\,(\partial_{xx} u + \partial_{yy} u)) + \partial_{xy}(\nu\,(\partial_{xx} u + \partial_{yy} u)) + \partial_{yy}(\nu\,(\partial_{xx} u + \partial_{yy} u))] + O(h^3), \tag{III.16}$$

so that

$$\mathcal{T}_j^{\text{adv-diff}} = \mathcal{T}_j^{\text{adv}} + \mathcal{T}_j^{\text{diff}} = \frac{h^2}{12}\,[\partial_{xx} r + \partial_{xy} r + \partial_{yy} r] + O(h^3), \tag{III.17}$$

where

$$r = \partial_x f + \partial_y g - \nu\,(\partial_{xx} u + \partial_{yy} u), \tag{III.18}$$

and thereby the second-order error term vanishes for $r = 0$, i.e., in the steady state. We emphasize that there are two requirements for constructing a uniformly third-order advection-diffusion scheme. First, the diffusion scheme must have a second-order error term that vanishes in the steady state. Second, the second order error must be in the form compatible with that of the advection scheme.

The linear Galerkin scheme (i.e., the continuous $P_1$ Galerkin scheme), which is equivalent to the three-point central finite-difference scheme on the grid considered here,[15] has a second-order error term,

$$\mathcal{T}_j^{\text{diff}} = \frac{\nu h^2}{12}(\partial_{xxxx} u + \partial_{yyyy} u) + O(h^3), \tag{III.19}$$

which does not vanish in the steady state, leading to

$$\begin{aligned} \mathcal{T}_j^{\text{adv-diff}} &= \mathcal{T}_j^{\text{adv}} + \mathcal{T}_j^{\text{diff}} \\ &= \frac{h^2}{12}\,[\partial_{xx}(\partial_x f + \partial_y g + \nu\partial_{xx} u) + \partial_{xy}(\partial_x f + \partial_y g) + \partial_{yy}(\partial_x f + \partial_y g + \nu\partial_{yy} u)] + O(h^3). \end{aligned}$$

Clearly, the second-order error term does not vanish. This scheme is, therefore, second-order accurate, and can be third-order accurate only in the advection limit ($\nu \to 0$). Without exploring various other diffusion schemes or seeking a general guiding principle, we took a third-order version of the linear Galerkin scheme described in Refs.16, 17, which is obtained by upgrading the element-gradient by a curvature correction. The curvature correction term is computed from the gradients reconstructed at nodes. See Ref.16 for details. Note that this third-order Galerkin scheme is a corrected linear Galerkin scheme, not a discontinuous Galerkin scheme nor the continuous $P_2$ Galerkin scheme.[16] This scheme has the following truncation error,[18]

$$\mathcal{T}_j^{\text{diff}} = \frac{h^2}{12}\,[\partial_{xx}(\nu\,(\partial_{xx} u + \partial_{yy} u)) + \partial_{xy}(\nu\,(\partial_{xx} u + \partial_{yy} u)) + \partial_{yy}(\nu\,(\partial_{xx} u + \partial_{yy} u))] + O(h^3), \tag{III.20}$$

and thus it is third-order accurate in the steady state where $\nu\,(\partial_{xx}u + \partial_{yy}u) = 0$. However, the sum of the third-order advection scheme and the third-order Galerkin scheme has the following truncation error:

$$\mathcal{T}_j^{\text{adv-diff}} = \mathcal{T}_j^{\text{adv}} + \mathcal{T}_j^{\text{diff}} = \frac{h^2}{12}\left[\partial_{xx}r' + \partial_{xy}r' + \partial_{yy}r'\right] + O(h^3), \tag{III.21}$$

where

$$r' = \partial_x f + \partial_y g + \nu\,(\partial_{xx}u + \partial_{yy}u). \tag{III.22}$$

We immediately notice that the diffusive term in $r'$ has a wrong sign, and therefore the second-order term does not vanish in the steady state where $r = 0$ but $r' \neq 0$. The scheme is only second-order accurate except in the advection limit or in the diffusion limit, i.e., not uniformly third-order accurate. This is the compatibility problem stated as the second requirement above. A similar discussion can be found in Ref.4, which pertains to second- and high-order residual-distribution schemes.

There is a possibility that the edge-based diffusion scheme in Refs.11,19 can also achieve third-order accuracy with a vanishing second-order error term. However, it may require a cubic fit rather than a quadratic fit for gradient reconstruction, and even if it is third-order accurate, it may not be compatible with the third-order advection scheme. Moreover, even if a compatible third-order diffusion scheme is found for regular grids, the same property is not necessarily guaranteed on irregular grids. While the search continues for a compatible third-order diffusion scheme, we show in the next section that the construction of uniformly third-order advection-diffusion schemes is *trivial* in the hyperbolic method. The compatibility problem does not exist because all terms are made hyperbolic and can be discretized in exactly the same way, thus yielding a fully compatible second-order error term.

### *III.F.2. Hyperbolic Formulation for Uniform Accuracy*

The construction of the third-order advection-diffusion scheme is made trivial by the fully hyperbolic formulation:[1]

$$\partial_t \mathbf{U} + \partial_x \mathbf{F} + \partial_y \mathbf{G} = \mathbf{0}, \tag{III.23}$$

where

$$\mathbf{F} = \mathbf{F}^a + \mathbf{F}^d + \mathbf{F}^s = \begin{bmatrix} au \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -\nu p \\ -u/T_r \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ (y - y_j)\,q/T_r \\ -(x - x_j)\,q/T_r \end{bmatrix}, \tag{III.24}$$

$$\mathbf{G} = \mathbf{G}^a + \mathbf{G}^d + \mathbf{G}^s = \begin{bmatrix} bu \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -\nu q \\ 0 \\ -u/T_r \end{bmatrix} + \begin{bmatrix} 0 \\ -(y - y_j)\,p/T_r \\ (x - x_j)\,p/T_r \end{bmatrix}, \tag{III.25}$$

where $(x_j, y_j)$ denotes the location of a node at which the system is discretized. Each part is hyperbolic and can be discretized by the same third-order upwind scheme. The numerical flux is constructed simply as a sum of the upwind advection flux, the upwind hyperbolic-diffusion flux, and the upwind source flux. The resulting scheme is guaranteed to be uniformly third-order accurate. Expanding the scheme on the regular triangular grid, we find the truncation error for each equation as

$$\begin{aligned} \mathcal{T}_j^u &= -\frac{\nu h}{6L_r}\left[(\sqrt{2}+\sqrt{5})\partial_x(p - \partial_x u) + \sqrt{2}\,\partial_y(p - \partial_x u) + \sqrt{2}\,\partial_x(q - \partial_y u) + (\sqrt{2}+\sqrt{5})\partial_y(q - \partial_y u)\right] \\ &+ \frac{h^2}{12}\left[\partial_{xx}r + \partial_{xy}r + \partial_{yy}r\right] + O(h^3), \end{aligned} \tag{III.26}$$

$$\mathcal{T}_j^p = -\frac{h^2}{6T_r}\left[(\partial_{xx} + \partial_{xy})(q - \partial_y u) + \partial_{xx}(p - \partial_x u) + \partial_y(\partial_x q - \partial_y p)\right] + O(h^3), \tag{III.27}$$

$$\mathcal{T}_j^q = -\frac{h^2}{6T_r}\left[(\partial_{xy} + \partial_{yy})(p - \partial_x u) + \partial_{yy}(q - \partial_y u) - \partial_x(\partial_x q - \partial_y p)\right] + O(h^3), \tag{III.28}$$

where $r = a\,\partial_x u + b\,\partial_y u - \nu(\partial_x p + \partial_y q)$. Observe that there are first- and second-order error terms but they all vanish in the steady state where $r = 0$, $p - \partial_x u = 0$, $q - \partial_y u = 0$, and $\partial_x q - \partial_y p = 0$. The most critical term is the second-order error term in the first equation, which vanishes in the steady state for the advection-diffusion equation, meaning that the advective and diffusive terms have been discretized in a perfectly compatible manner. The scheme is, therefore, uniformly third-order accurate for any set of parameters, $(a, b)$ and $\nu$. We expect it to be uniformly third-order accurate on unstructured triangular grids also because the third-order accuracy on unstructured triangular grids has already been demonstrated for hyperbolic systems in Refs.12, 13, 14. In short, whatever is true about the third-order scheme for hyperbolic systems is true for the fully hyperbolic advection-diffusion system.

For the third-order scheme, the gradient computation must be exact for quadratic functions, i.e., a quadratic fit. It requires five neighbors and may extend beyond the edge-connected neighbors in some cases. In the previous paper,[1] we selected 10 neighbors by including neighbors of the edge-connected neighbors as necessary, and stored the list of 10 neighbors at each node. Here, we avoid carrying information on the neighbors of the neighbors by implementing the quadratic gradient reconstruction in two steps, where each step is compact, as described in Appendix. This method involves all neighbors of the edge-connected neighbors. The total number of neighbors can be as large as 18, and it can be much more than necessary in many cases. A smart selection of a minimal number of neighbors may be possible, but in this study we employ the two-step method for robustness and simplicity. It is robust as it has far more neighbors than necessary even at boundary nodes, and simple as it can be implemented with the list of edge-connected neighbors only.

## IV. Numerical Results

We consider the steady advection-diffusion problem in a square domain with the exact solution given by[4]

$$u(x, y) = \cos(2\pi\eta) \exp\left(\frac{-2\pi^2\nu}{1 + \sqrt{1 + 4\pi^2\nu^2}}\,\xi\right), \tag{IV.1}$$

where $\xi = ax + by$, $\eta = bx - ay$, and with the Dirichlet boundary condition. The advection vector is set as $(a, b) = (1.23, 0.12)$ and $\nu$ is determined from the parameter $Re$ by

$$\nu = \frac{\sqrt{a^2 + b^2}}{Re}, \tag{IV.2}$$

for $Re = 10^{-6}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3, 10^6$. Numerical results are presented for a series of independently generated eight irregular triangular grids with $N$ nodes, where $N$ =2048, 8192, 18432, 32768, 51200, 73728, 100352, 131072. The coarsest grid is shown in Figure 2. As can be seen, the grid is fully irregular with random number of neighbors and some vanishingly small volumes. The first, second, and third order hyperbolic advection-diffusion schemes, designated as SchemeII(1st), SchemeII(2nd), and SchemeII(3rd), are compared with two traditional schemes. One is the third-order advection scheme with the linear Galerkin scheme, and the other with the third-order Galerkin scheme considered in Section III.F.1. The former is designated as Galerkin, and the latter as Galerkin(3rd). These schemes are termed 'traditional' because they are scalar schemes directly solving the advection-diffusion equation (II.1). The implicit iterative method is implemented with the first-order residual Jacobian for the advective part and the exact linear-Galerkin Jacobian for the diffusive part for both traditional schemes. For the Galerkin(3rd) scheme, the Jacobian for the diffusive part is, therefore, approximate. See Table 1 for a summary of discretizations and Jacobians. In all cases, the GS relaxation is terminated when the residual of the linear system is reduced by two orders of magnitude in the $L_1$ norm. For gradient reconstruction, the unweighted linear LSQ method is used for SchemeII(2nd), and the two-step quadratic LSQ method as in Appendix is used for SchemeII(3rd) and the two traditional schemes. In all cases, the initial solution is set by the exact solution randomly perturbed. Steady convergence is taken to be reached when the residual in the $L_1$ norm drops by ten orders of magnitude or reaches the machine zero.

Error convergence results are shown in Figures 3 to 10. Only the results for $u$ and $p$ are shown because the results for $q$ is very similar to those for $p$. Also, the results for $Re = 10^{-3}$ are not shown because they look identical to those for $Re = 10^{-6}$. For the traditional schemes, $p$ corresponds to the $x$-component of the quadratic LSQ gradients. First, it is observed that the Galerkin scheme is second-order accurate in the solution and first-order accurate in the gradients (even with the quadratic LSQ fit) except in the advection limit $Re = 10^6$ (Figure 10) where the third-order advection scheme dominates and yields third- and second-order accuracy in the solution and the gradients, respectively. Second, the Galerkin(3rd) scheme gives third-order accuracy in the solution and second-order accuracy in the gradients in the diffusion dominated cases, but the

accuracy begins to deteriorate by one order at $Re = 1$ (Figure 6) and starts recovering third-order accuracy at $Re = 10^3$ (Figure 9). As predicted, therefore, the scheme is not uniformly third-order accurate. On the other hand, SchemeII(1st), SchemeII(2nd), and SchemeII(3rd) never encounter such accuracy deterioration. As results show, SchemeII(1st) and SchemeII(2nd) are uniformly first- and second-order accurate, respectively, in both the solution and the gradients. Observe also that as advection dominates ($Re = 10^3, 10^6$), SchemeII(1st) and SchemeII(2nd) yield one order higher accuracy in the solution, i.e., second-order and third-order accuracy, respectively. The second/third-order accuracy is expected because the gradients are computed with first/second-order accuracy, which is employed in the face-reconstruction for the advection scheme. This way of achieving second/third-order accuracy for advection is interesting and economical: second-order advection scheme with no gradient computations, and third-order advection scheme with linear (not quadratic) LSQ gradients. In effect, the hyperbolic diffusion scheme here plays a single role of providing accurate gradients with the diffusive term kept negligibly small. Finally, observe that SchemeII(3rd) gives third-order accuracy for both the solution and the gradients for all values of $Re$. It is worth emphasizing that the gradients are third-order accurate even in the advection limit. The third-order scheme is very efficient in terms of the accuracy in the gradients as we shall discuss next.

Iterative convergence results are presented in Figures 11 to 18. The same legend applies as in the error convergence plots, and therefore it is not shown. Again, the results for $Re = 10^{-3}$ are not shown because they look identical to those for $Re = 10^{-6}$. In each figure, three plots are shown: total iterations for convergence, the total number of the GS relaxations, and the total CPU time taken for convergence, all versus $1/h$, where $h = 1/\sqrt{N}$. First, it is observed that the convergence characteristics of the Galerkin(3rd) scheme are nearly identical to those of the Galerkin scheme. This is an unexpected result because the Jacobian is exact for the Galerkin scheme but only approximate for the Galerkin(3rd) scheme. The results imply that the Jacobian of the linear Galekrin scheme behaves like exact for the third-order Galerkin scheme. Second, the number of GS sweeps increases quadratically as the grid gets finer in the diffusion dominated cases while increases linearly in the advection dominated cases. It is consistent with the change in the condition number of the Jacobian matrix from $O(1/h^2)$ in the diffusion limit to $O(1/h)$ in the advection limit. In terms of CPU time, as shown in the right-most plot, the cost of these traditional schemes varies from $O(1/h^4)$ or equivalently $O(N^2)$ in the diffusion limit to $O(1/h^3)$ or $O(N^{1.5})$ in the advection limit. On the other hand, the hyperbolic schemes preserve, *for all values of Re*, $O(N^{1.5})$ convergence in the CPU time. This result is a direct consequence of solving the hyperbolic advection-diffusion system instead of the scalar advection-diffusion quation. In the diffusion limit, the hyperbolic schemes are, therefore, $O(1/h)$ times faster than the traditional schemes. Note that the acceleration factor grows as the grid gets finer (i.e., as $h \to 0$). In the advection limit, both traditional and hyperbolic schemes show $O(N^{1.5})$ convergence in the CPU time. Results for $Re = 10^6$ show that the traditional schemes are (2 or 3 times) faster than the hyperbolic schemes, which is expected because the hyperbolic schemes solve two additional equations. However, it is not immediately clear if the traditional schemes are more efficient because the third-order hyperbolic scheme is capable of delivering third-order accurate gradients. If we focus on the accuracy in the gradients, the third-order hyperbolic scheme is to be compared with a fourth-order scalar advection-diffusion scheme. The cost of the third-order hyperbolic scheme being comparable with that of the second-order scheme, it implies a tremendous potential advantage of the third-order hyperbolic scheme for applications where accurate gradients are sought. Note also that SchemeII(2nd) achieves third-order accuracy in the advection limit with the linear LSQ gradients, not the quadratic LSQ gradients.

Table 2 shows how the number of iterations and GS sweeps vary with $Re$ for the finest grid. For the traditional schemes, the number of GS sweeps increases significantly as diffusion dominates, while the the number of iterations increases as advection dominates (due to the approximate Jacobian). For the hyperbolic schemes, the number of iterations does not vary significantly, but the number of GS sweeps increases as diffusion dominates. If desired, it may be possible to reduce the number of GS sweeps by deriving an optimal $L_r$ for the numerical scheme in the diffusion limit as in Ref.2, not for the differential equations as in Refs.1, 7. Yet, we emphasize that the hyperbolic schemes are already an order-of-magnitude more efficient than the traditional schemes in the diffusion limit. See table 3 for the CPU time comparison for the finest grid.

## V. Concluding Remarks

We have extended the diffusion schemes developed in the previous paper[1] to the advection-diffusion equation, generating uniformly accurate first, second, and third order advection-diiffusion schemes on unstructured triangular grids. The advective, diffusive, and source terms have been discretized in a unified framework by the method of hyperbolic systems that converts the diffusive and source terms into hyperbolic systems. The devel-

oped hyperbolic schemes are node-centered edge-based finite-volume schemes with the upwind flux for all terms. An implicit iterative method has been developed for all schemes based on the exact Jacobian of the first-order scheme. Also, a two-step implementation of the quadratic LSQ gradient reconstruction has been proposed for robustness and simplicity, in which each step is compact, requiring only the list of the edge-connected neighbors.

The developed schemes were compared with two traditional schemes: the third-order advection scheme with the linear Galerkin diffusion scheme, and with a third-order version of the linear Galerkin diffusion scheme.[16,17] It was shown analytically as well as numerically that the latter scheme cannot be third-order accurate when advection and diffusion are equally important ($Re = 1, 10, 10^2$ in Figures 6, 7, 8) while the former is third-order accurate only in the advection limit. Typically, the accuracy is deteriorated by one order, and in a critical case ($Re = 100$), the deterioration begins to appear on fine grids. We emphasize again that the third-order Galerkin diffusion scheme is a corrected version of the linear Galerkin diffusion scheme as described in Ref.16, not the discontinuous Galerkin scheme nor the continuous $P_2$ Galerkin scheme. On the other hand, the developed schemes have been confirmed to be uniformly accurate up to the design (or higher) order accuracy for all values of $Re$, i.e., $Re = 10^{-6}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3, 10^6$. Specifically, the first-order scheme has been shown to yield first-order accurate solution and gradients, and second-order accurate solution in the advection limit. The second-order scheme has been shown to yield second-order accurate solution and gradients, and third-order accurate solution in the advection limit. The third-order scheme has been shown to yield uniformly third-order accurate solution and gradients.

For iterative convergence, the hyperbolic schemes have been shown to bring $O(1/h)$ acceleration in convergence over traditional schemes in the diffusion limit. No such acceleration was observed in the advection limit as expected, and the developed schemes converged somewhat slower than the traditional schemes in the test problem considered. However, there are decisive advantages in the hyperbolic schemes, which overwhelm the slightly slower convergence in the advection limit. First, the second-order scheme delivers third-order accurate solutions and second-order accurate gradients with linear LSQ gradients (not with quadratic LSQ gradients). Second, the third-order scheme produces third-order accurate gradients, which typically requires fourth-order schemes. Also noticeable is that the first-order scheme yields second-order accurate solutions and first-order accurate gradients in a compact stencil (no gradient reconstruction) at a fast Newton-like convergence with the exact linearization.

A particularly important contribution of this paper is the demonstration of the simplified approach: construct an advection-diffusion scheme as a sum of an advection scheme and a hyperbolic diffusion scheme. The approach has been shown to work well in terms of accuracy as well as efficiency for a wide range of parameter $Re$ from the diffusion limit ($Re = 10^{-6}$) to the advection limit ($Re = 10^6$). For the compressible Navier-Stokes equations, it dramatically simplifies the construction of numerical schemes as a viscous scheme can be developed independently from the inviscid scheme. As shown in Ref.8, the viscous terms can be made a hyperbolic system, its eigen-structure can be fully analyzed, and therefore discretized easily by the upwind scheme or any other scheme suitable for hyperbolic systems. It can then be added to any inviscid scheme to construct a hyperbolic Navier-Stokes scheme.

Finally, we emphasize again that the core idea of the hyperbolic method lies in the construction of a first-order hyperbolic system for target differential equations, and therefore it is generally applicable to any high-order partial differential equation as well as any discretization method. Applications to other types of equations are now made particularly simple as we have shown in this paper that numerical schemes can be constructed for each non-hyperbolic term independently by turning it into a hyperbolic system. Opportunities are still wide open.

## References


[1]Nishikawa, H., "First, Second, and Third Order Finite-Volume Schemes for Diffusion," *51st AIAA Aerospace Sciences Meeting*, AIAA Paper 2011-1125, Grapevine, Texas, January 2013.

[2]Nishikawa, H., "A First-Order System Approach for Diffusion Equation. I: Second-Order Residual-Distribution Schemes," *Journal of Computational Physics*, Vol. 227, 2007, pp. 315–352.

[3]Corre, C. and Du, X., "A Residual-Based Scheme for Computing Compressible Flows on Unstructured Grids," *Computers and Fluids*, Vol. 38, 2009, pp. 1338–1347.

[4]Nishikawa, H. and Roe, P. L., "On High-Order Fluctuation-Splitting Schemes for Navier-Stokes Equations," *Computational Fluid Dynamics 2004*, edited by C. Groth and D. W. Zingg, Springer-Verlag, 2004, pp. 799–804.

[5]Ricchiuto, M., Villedieu, N., Abgrall, R., and Deconinck, H., "On Uniformly High-Order Accurate Residual Distribution Schemes for Advection-Diffusion," *Journal of Computational and Applied Mathematics*, Vol. 215, 2008, pp. 547–556.

[6]Nishikawa, H., "Divergence Formulation of Source Term," *Journal of Computational Physics*, Vol. 231, 2012, pp. 6393–6400.

[7]Nishikawa, H., "A First-Order System Approach for Diffusion Equation. II: Unification of Advection and Diffusion," *Journal of Computational Physics*, Vol. 229, 2010, pp. 3989–4016.

[8]Nishikawa, H., "New-Generation Hyperbolic Navier-Stokes Schemes: $O(1/h)$ Speed-Up and Accurate Viscous/Heat Fluxes," *20th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2011-3043, Hawaii, 2011.

[9]Lowrie, R. B. and Morel, J. E., "Methods for Hyperbolic Systems with Stiff Relaxation," *International Journal for Numerical Methods in Fluids*, Vol. 40, 2002, pp. 413–423.

[10]Gómez, H., Colominas, I., Navarrina, F., and Casteleiro, M., "A Discontinuous Galerkin Method for a Hyperbolic Model for Convection-Diffusion Problems in CFD," *International Journal for Numerical Methods in Engineering*, Vol. 71, 2007, pp. 1342–1364.

[11]Nishikawa, H., "Beyond Interface Gradient: A General Principle for Constructing Diffusion Schemes," *40th AIAA Fluid Dynamics Conference and Exhibit*, AIAA Paper 2010-5093, Chicago, 2010.

[12]Katz, A. and Sankaran, V., "Mesh quality effects on the accuracy of CFD solutions on unstructured meshes," *Journal of Computational Physics*, Vol. 230, 2011, pp. 7670–7686.

[13]Katz, A. and Sankaran, V., "An Efficient Correction Method to Obtain a Formally Third-Order Accurate Flow Solver for Node-Centered Unstructured Grids," *Journal of Scientific Computing*, Vol. 51, 2012, pp. 375–393.

[14]Diskin, B. and Thomas, J. L., "Effects of mesh regularity on accuracy of finite-volume schemes," *50th AIAA Aerospace Sciences Meeting*, AIAA Paper 2012-0609, Nashville, Tennessee, 2012.

[15]Diskin, B., Thomas, J. L., Nielsen, E. J., Nishikawa, H., and White, J. A., "Comparison of Node-Centered and Cell-Centered Unstructured Finite-Volume Discretizations: Viscous Fluxes," *AIAA Journal*, Vol. 48, No. 7, July 2010, pp. 1326–1338.

[16]Nishikawa, H., "Higher-Order Discretization of Diffusion Terms in Residual-Distribution Methods," *VKI Lecture Series 2006-01: CFD - High Order Discretization Methods*, edited by H. Deconinck and M. Ricchiuto, von Karman Institute for Fluid Dynamics, Belgium, 2006.

[17]Nishikawa, H., "Multigrid Third-Order Least-Squares Solution of Cauchy-Riemann Equations on Unstructured Triangular Grids," *International Journal for Numerical Methods in Fluids*, Vol. 53, 2007, pp. 443–454.

[18]Nishikawa, H. and Roe, P. L., "High-Order Viscous Discretization on Unstructured Grids," Unpublished, `http://cfdnotes.com/cfdnotes_high_diffusion.pdf`, 2004.

[19]Nishikawa, H., "Robust and Accurate Viscous Discretization via Upwind Scheme - I: Basic Principle," *Computers and Fluids*, Vol. 49, No. 1, 2011, pp. 62–86.

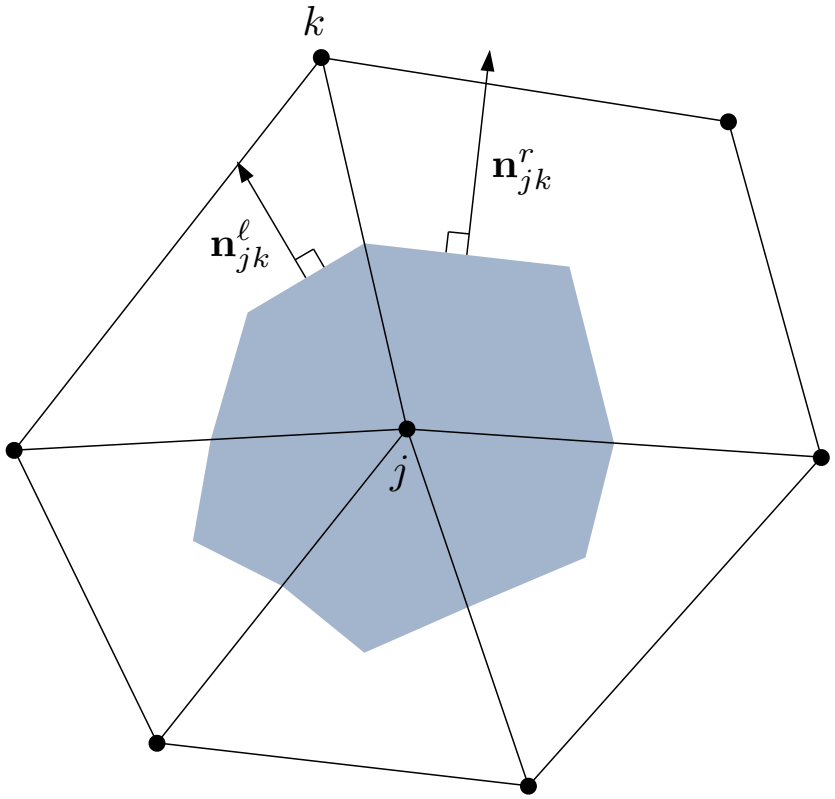


**Figure 1. Dual control volume for the node-centered finite-volume method with scaled outward normals associated with an edge, $\{j, k\}$.**

| Scheme | Discretization | | Jacobian | |
|---|---|---|---|---|
| | Advection | Diffusion | Advection | Diffusion |
| Galerkin | Third-order upwind | Linear Galerkin | First-order upwind | Exact |
| Galerkin(3rd) | Third-order upwind | Third-order Galerkin | First-order upwind | Linear Galerkin |
| SchemeII(1st) | First-order upwind | | Exact | |
| SchemeII(2nd) | Second-order upwind | | First-order upwind | |
| SchemeII(3rd) | Third-order upwind | | First-order upwind | |

**Table 1. Summary of discretizations and Jacobians.**

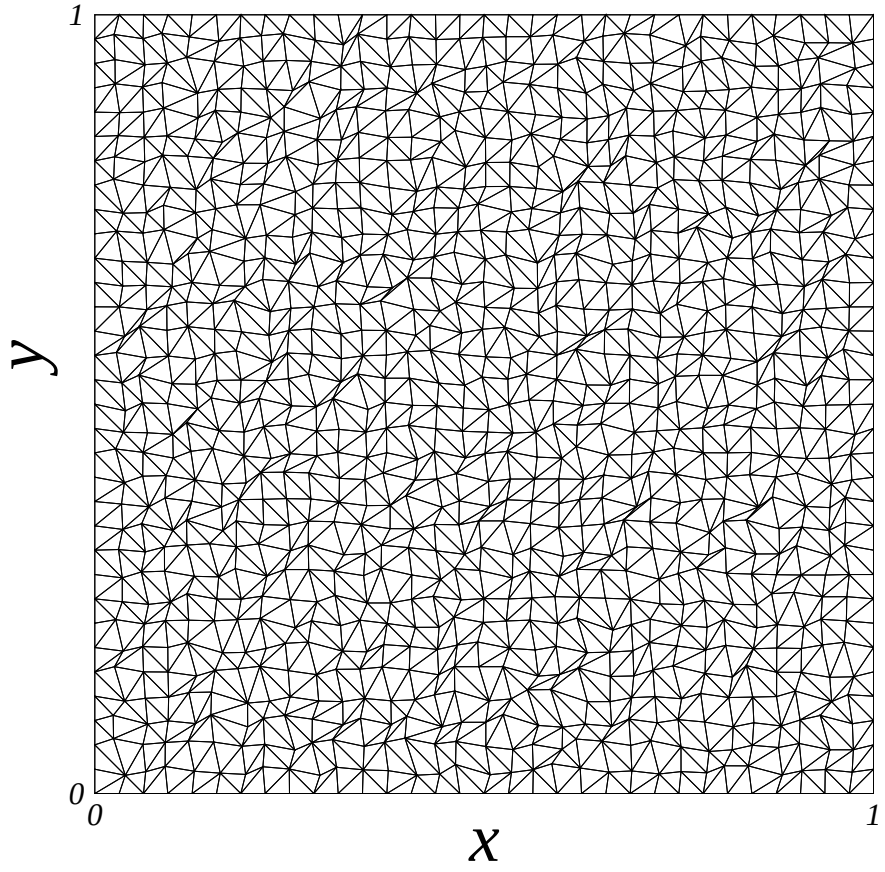


**Figure 2. Irregular triangular grid with 2048 nodes.**

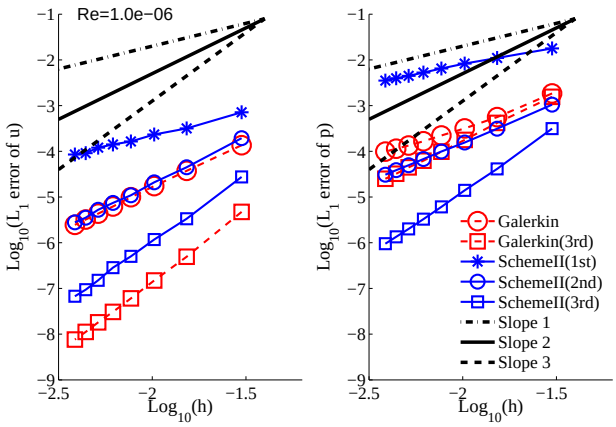


**Figure 3. Case $Re = 10^{-6}$**

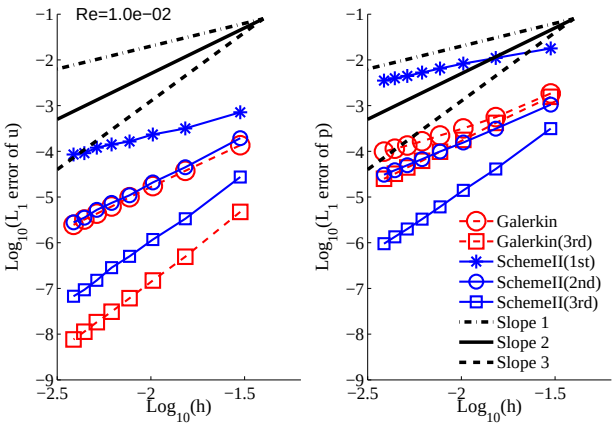


**Figure 4. Case $Re = 10^{-2}$**

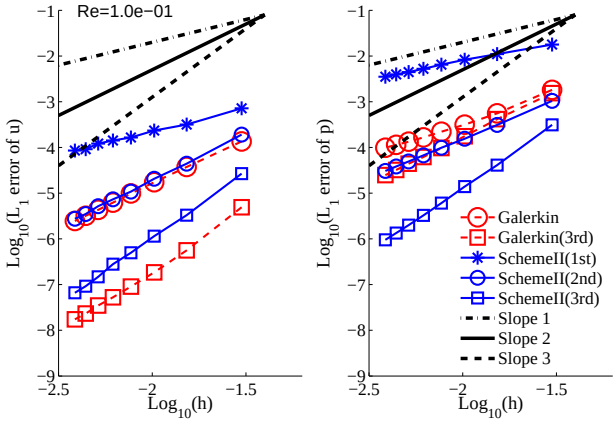


**Figure 5. Case $Re = 10^{-1}$**

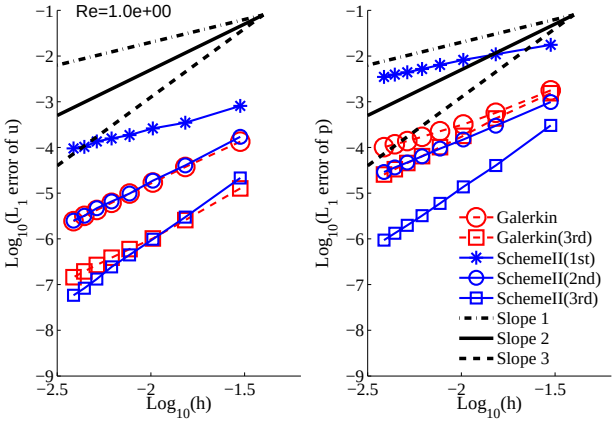


**Figure 6. Case $Re = 1$**

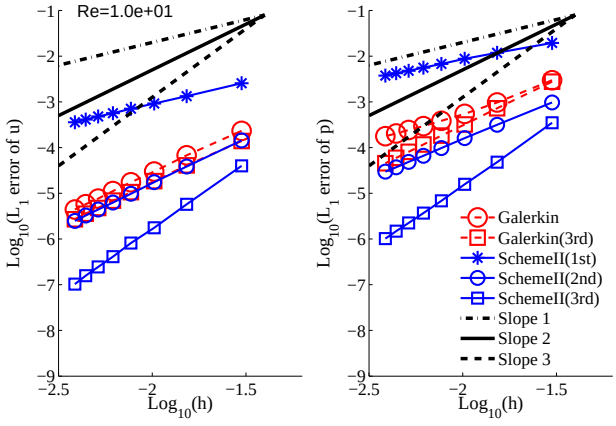


**Figure 7. Case $Re = 10$**

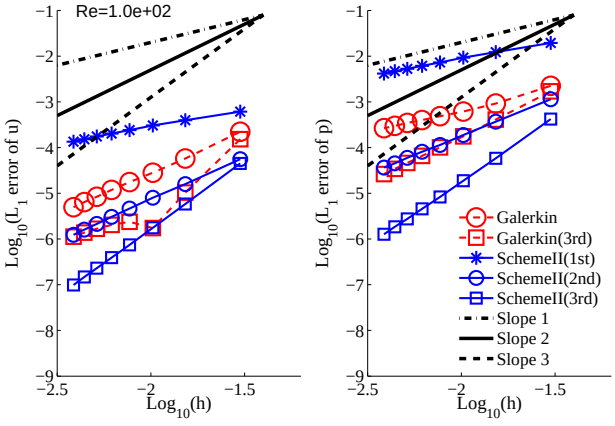


**Figure 8. Case $Re = 10^2$**

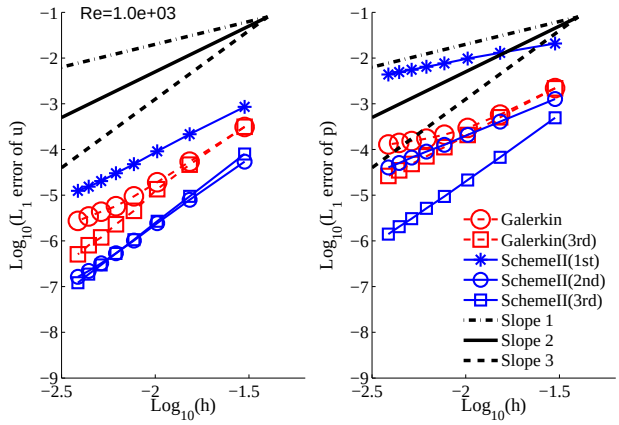


**Figure 9. Case $Re = 10^3$**

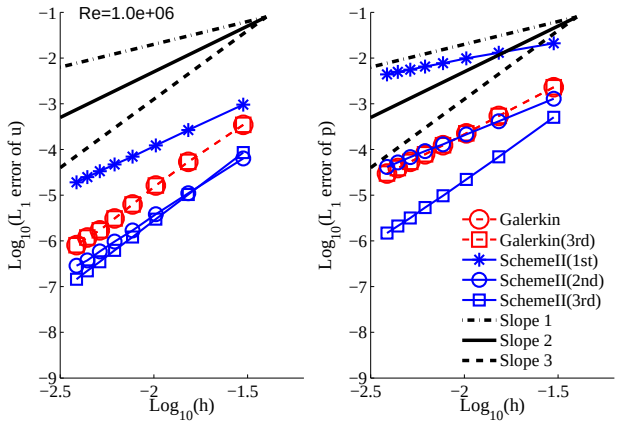


**Figure 10. Case $Re = 10^6$**

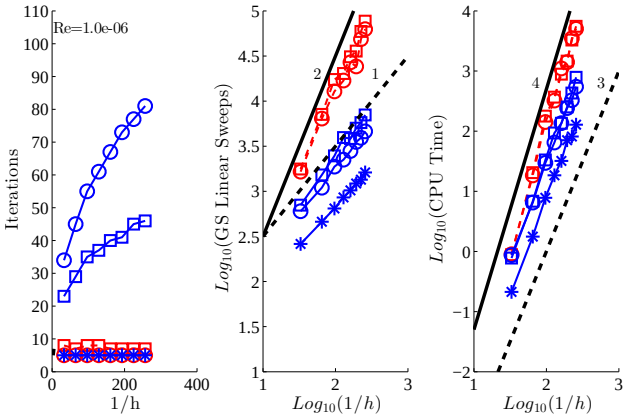


**Figure 11. Case $Re = 10^{-6}$**

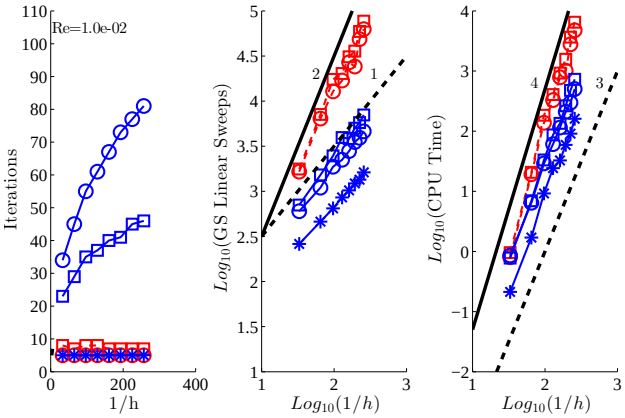


**Figure 12. Case $Re = 10^{-2}$**

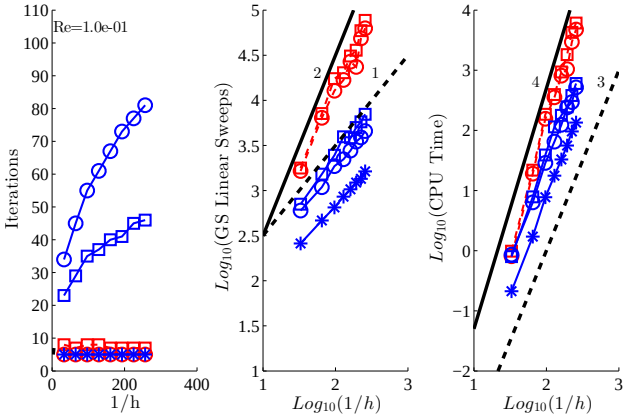


**Figure 13. Case $Re = 10^{-1}$**

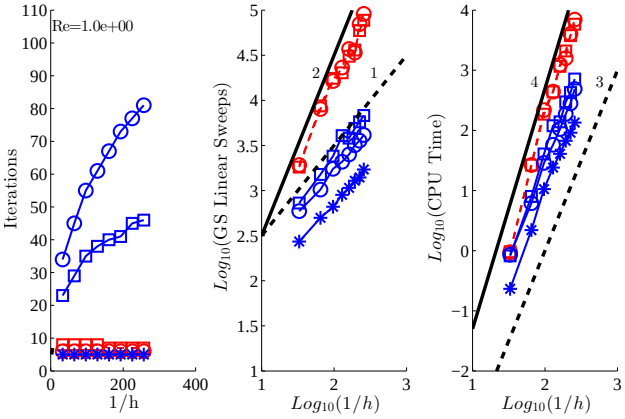


**Figure 14. Case $Re = 1$**

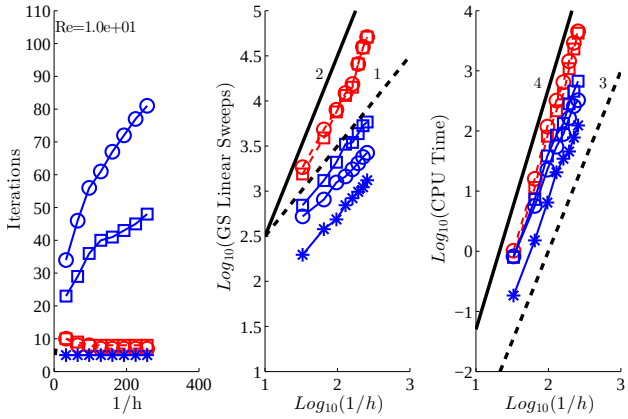


**Figure 15. Case $Re = 10$**

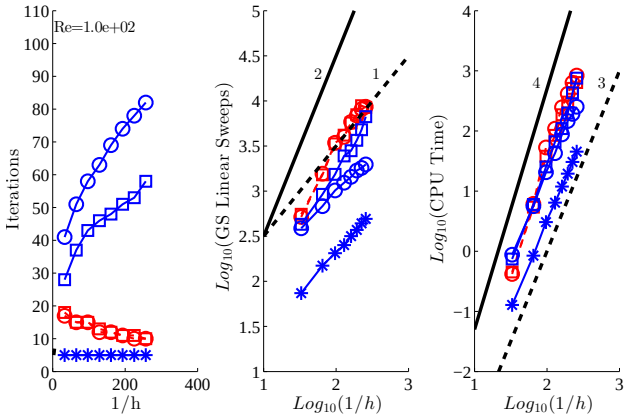


**Figure 16. Case $Re = 10^2$**

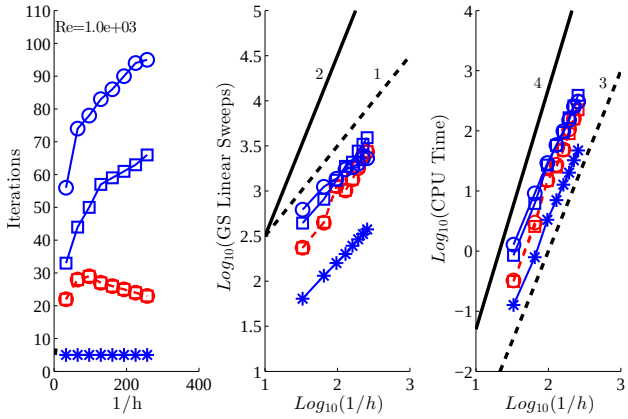


**Figure 17. Case $Re = 10^3$**

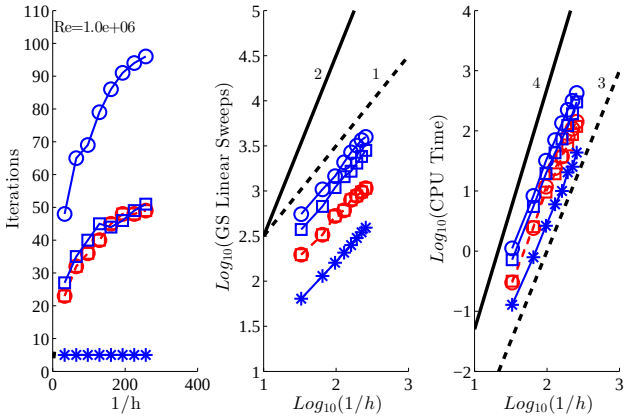


**Figure 18. Case $Re = 10^6$**

| Scheme | $Re = \sqrt{a^2+b^2}/\nu$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | $10^{-6}$ | $10^{-3}$ | $10^{-2}$ | $10^{-1}$ | 1 | 10 | $10^2$ | $10^3$ | $10^6$ |
| Galerkin | 5(12564) | 5(12564) | 5(12565) | 5(12567) | 6(15124) | 7(7292) | 10(861) | 23(118) | 49(22) |
| Galerkin(3rd) | 7(10987) | 7(10987) | 7(10989) | 7(11003) | 7(11018) | 8(6385) | 10(848) | 23(119) | 49(22) |
| SchemeII(1st) | 5(324) | 5(324) | 5(324) | 5(326) | 5(343) | 5(264) | 5(98) | 5(75) | 5(78) |
| SchemeII(2nd) | 77(58) | 77(58) | 77(58) | 77(57) | 77(52) | 77(34) | 78(25) | 93(24) | 94(42) |
| SchemeII(3rd) | 46(151) | 46(151) | 46(151) | 46(151) | 46(146) | 46(122) | 58(116) | 66(59) | 51(56) |

**Table 2. Total number of iterations in the finest grid case. The number in the parenthesis is the average number of GS-sweeps per iteration required to ensure two orders of magnitude reduction in the residual of the linear system.**

| Scheme | $Re = \sqrt{a^2+b^2}/\nu$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | $10^{-6}$ | $10^{-3}$ | $10^{-2}$ | $10^{-1}$ | 1 | 10 | $10^2$ | $10^3$ | $10^6$ |
| Galerkin | 5.0e+03 | 4.8e+03 | 4.8e+03 | 4.8e+03 | 7.0e+03 | 4.5e+03 | 8.3e+02 | 3.1e+02 | 1.4e+02 |
| Galerkin(3rd) | 5.5e+03 | 5.6e+03 | 6.5e+03 | 6.1e+03 | 5.8e+03 | 4.2e+03 | 6.4e+02 | 2.2e+02 | 1.2e+02 |
| SchemeII(1st) | 1.4e+02 | 1.3e+02 | 1.3e+02 | 1.3e+02 | 1.4e+02 | 1.1e+02 | 4.3e+01 | 3.9e+01 | 3.6e+01 |
| SchemeII(2nd) | 5.0e+02 | 4.7e+02 | 5.2e+02 | 4.4e+02 | 4.0e+02 | 2.9e+02 | 2.5e+02 | 2.8e+02 | 4.2e+02 |
| SchemeII(3rd) | 7.2e+02 | 7.2e+02 | 7.1e+02 | 6.4e+02 | 5.8e+02 | 5.3e+02 | 5.9e+02 | 3.9e+02 | 3.0e+02 |

**Table 3. Total CPU time for convergence in the finest grid case.**

## Appendix: Two-Step Implementation of Quadratic LSQ Gradient Reconstruction

First, we compute and store the coefficients for the gradient once for a given grid.

1. *Construct temporary data:*

   For each node $j \in \{J\}$, loop over the edge-connected neighbors $k \in \{k_j\}$ and store the edge-vector:

   $$\Delta \mathbf{x}_j(k) = \mathbf{x}_k - \mathbf{x}_j, \tag{V.1}$$

   where $\Delta \mathbf{x}_j(k) = (\Delta x_j(k), \Delta y_j(k))$, $\mathbf{x}_k = (x_k, y_k)$, and $\mathbf{x}_j = (x_j, y_j)$. One can skip this step if the data is already available in a code.

2. *Compute gradient coefficients:*

   For each node $j \in \{J\}$ , loop over the neighbors of $k$, $\ell \in \{\ell_k\}$ within the loop over the edge-connected neighbors $k \in \{k_j\}$:

   $$\Delta \mathbf{x} = \Delta \mathbf{x}_j(k) + \Delta \mathbf{x}_k(\ell), \tag{V.2}$$

   and accumulate each entry of a 5×5 LSQ matrix for a quadratic fit, $\mathbf{A}_{LSQ}$. Note that $\Delta \mathbf{x} = 0$ if the neighbor coincides with the node $j$. It is not necessary, but we reset $\Delta \mathbf{x} = \Delta \mathbf{x}_j(k)$ in that case to increase the contribution from the edge-connected neighbors. After the loop over $k \in \{k_j\}$, compute the inverse of the LSQ matrix, $\mathbf{A}^{-1}_{LSQ}$. Next, set $i = 0$ and repeat the same double loop. Within the double loop, increment $i$ by $i = i + 1$, and compute the coefficients:

   $$\mathbf{c}_{ji} = \mathbf{A}^{-1}_{LSQ}\mathbf{b}, \tag{V.3}$$

   where $\mathbf{c}_{ji} = (c^x_{ji}, c^y_{ji}, c^{xx}_{ji}, c^{xy}_{ji}, c^{yy}_{ji})$, $\mathbf{b} = (\Delta x, \Delta y, \Delta x^2/2, \Delta x \Delta y, \Delta y^2/2)$. Save the first two components, $c^x_{ji}$ and $c^y_{ji}$, at the node $j$.

The remaining coefficients, $c^{xx}_{ji}$, $c^{xy}_{ji}$, and $c^{yy}_{ji}$, can be used to compute the second derivatives but not required in the third-order scheme. Note that the method automatically introduces slight weights based on the connectivities, e.g., 2 if a node is processed twice, which happens for the edge-connected neighbor shared by two adjacent elements. We point out that the only data needed are the coefficients $c^x_{ji}$ and $c^x_{ji}$ for each node, and other data such as $\Delta x_j$ and $\Delta y_j$ computed at the first step as well as the LSQ matrices can be deleted at the end of the process.

Having computed and stored the coefficients, we can perform the gradient reconstruction at every residual evaluation in two steps. We outline the procedure for the variable $p$, but it is equally valid for any variable.

1. *Construct temporary data:* For each node $j \in \{J\}$ , loop over the edge-connected neighbors $k \in \{k_j\}$ and store the edge-difference of the variable for which the gradient is sought:

   $$\Delta p_j(k) = p_k - p_j. \tag{V.4}$$

2. *Compute the gradient:* For each node $j \in \{J\}$ , initialize the gradient, $\nabla p_j = 0$ and the counter, $i = 0$, and loop over the neighbors of $k$, $\ell \in \{\ell_k\}$ within the loop over the edge-connected neighbors $k \in \{k_j\}$. Within the double loop, increment $i$ by $i = i + 1$ and accumulate the gradient contribution:

$$\nabla p_j = \nabla p_j + \Delta p \begin{bmatrix} c_{ji}^x \\ c_{ji}^y \end{bmatrix}, \quad \Delta p = \Delta p_j(k) + \Delta p_k(\ell), \tag{V.5}$$

or $\Delta p = \Delta p_j(k)$ in the case the neighbor coincides with $j$ if the reset has been employed in the calculation of the coefficients in Equation (V.2).

## Acknowledgments

This work has been funded by the U.S. Army Research Office under the contract/grant number W911NF-12-1-0154 with Dr. Frederick Ferguson as the program manager. Support by Software CRADLE and NASA Langley Research Center is greatly acknowledged.